

Exercise 06.1:CodePipeline :Deploying a sample webapp using CodePipeline

Step 1: Start your EC2 instances with tags Dev and Production

Step 2: Click CodePipeline

Developer Tools

CodePipeline

- ▶ Source • CodeCommit
- ▶ Build • CodeBuild
- ▶ Deploy • CodeDeploy
- ▶ Pipeline • CodePipeline
 - Getting started**
 - Pipelines
 - Settings

Go to resource

Feedback

AWS CodePipeline

visualize and automate the different stages of your software release process

AWS CodePipeline is a continuous integration and continuous delivery service for fast and reliable application and infrastructure updates. CodePipeline builds, tests, and deploys your code every time there is a code change, based on the release process models you define..

Create AWS CodePipeline pipeline

Get started with AWS CodePipeline by creating your first continuous delivery and continuous integration pipeline.

Create pipeline

Step 3: Click Create Pipeline

Step 1
Choose pipeline settings

Step 2
Add source stage

Step 3
Add build stage

Step 4
Add deploy stage

Step 5
Review

Choose pipeline settings Info

Pipeline settings

Pipeline name
Enter the pipeline name. You cannot edit the pipeline name after it is created.
No more than 100 characters

Service role

New service role
Create a service role in your account

Existing service role
Choose an existing service role from your account

Role name

Type your service role name

Allow AWS CodePipeline to create a service role so it can be used with this new pipeline

Step 4: Give the following details

Pipeline name: my-webpage-pipeline

Service role: Default

Advanced Settings:

Custom Location: Create a bucket and choose it.

Encryption Key: Default

Click Next

Developer Tools > CodePipeline > Pipelines > Create new pipeline

Step 1 Choose pipeline settings

Step 2 Add source stage

Step 3 Add build stage

Step 4 Add deploy stage

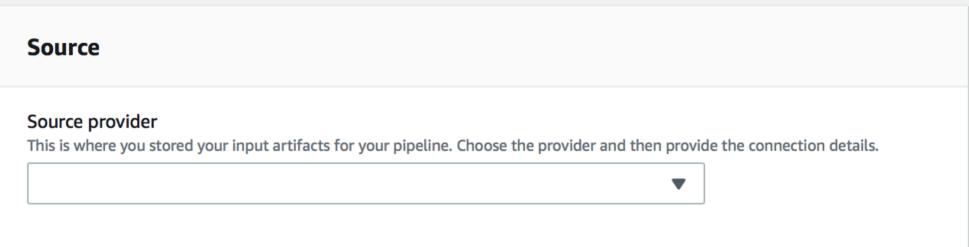
Step 5 Review

Add source stage Info

Source

Source provider
This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.

Cancel Previous Next



Step 5: Select CodeCommit, Repository name and Branch , Change DetectionOptions=Default click Next

Step 6: Click Skip build stage.

Developer Tools > CodePipeline > Pipelines > Create new pipeline

Step 1 Choose pipeline settings

Step 2 Add source stage

Step 3 Add build stage

Step 4 Add deploy stage

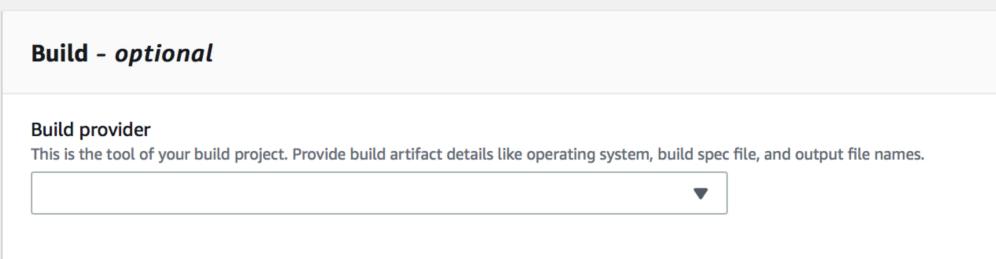
Step 5 Review

Add build stage Info

Build - optional

Build provider
This is the tool of your build project. Provide build artifact details like operating system, build spec file, and output file names.

Cancel Previous Skip build stage Next



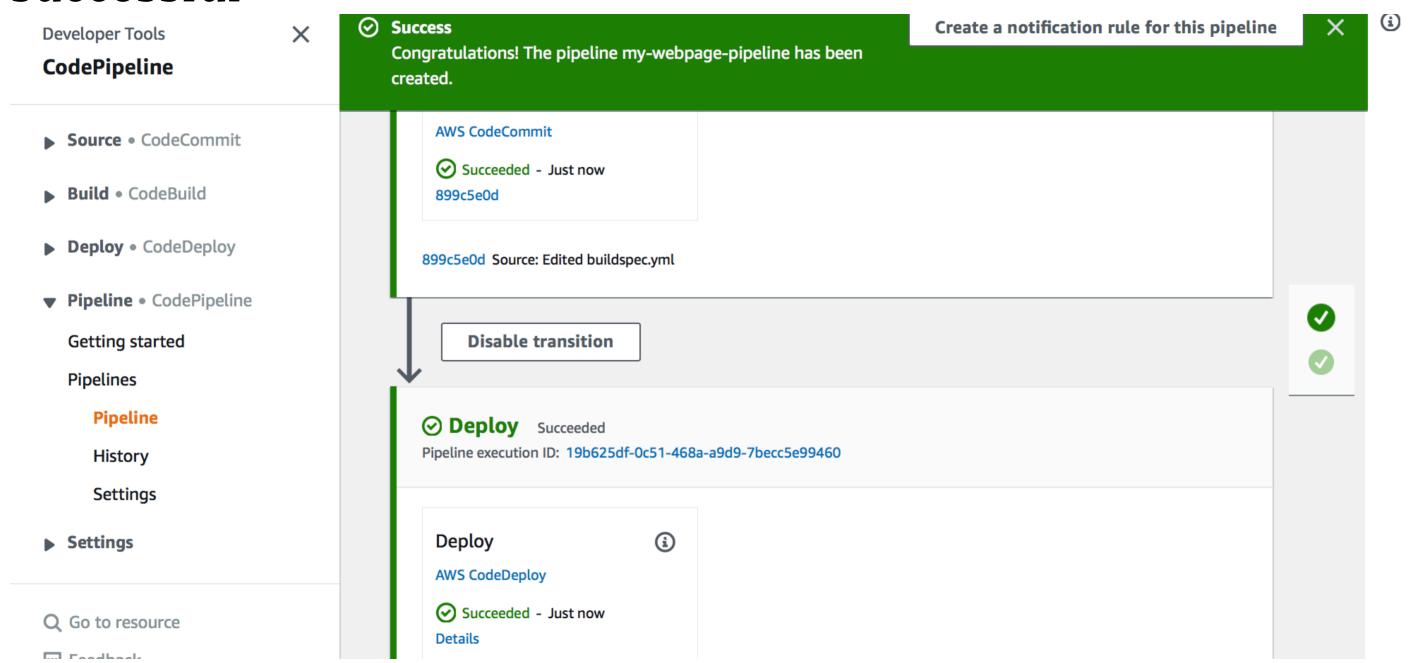
Step 7: Select Code Deploy,Region ,App name and deployment grp (Dev) and click Next

The screenshot shows the 'Deploy' step configuration screen in the AWS CodePipeline console. On the left, a sidebar lists steps: Step 4 (Add deploy stage), Step 5 (Review), and Step 6 (Next). The main area is titled 'Deploy' and contains the following fields:

- Deploy provider:** AWS CodeDeploy (selected)
- Region:** Asia Pacific (Mumbai)
- Application name:** my-deploy-app
- Deployment group:** my-qa-dep-grp

At the bottom right are 'Cancel', 'Previous', and 'Next' buttons, with 'Next' being highlighted.

Step 8: Review the details and click Create Pipeline. You should see a successful completion screen as follows if everything went successful



Step 9: Go to the browser and type the Public IP to get a congratulations page which confirms the successful deployment of your webapp using codepipeline

Congratulations

This application was deployed on i-0463e4e345f6af4f8 in ap-south-1a using AWS CodeDeploy.

For next steps, read the [AWS CodeDeploy Documentation](#).

Step 10: Make some changes in the index.html and commit the code. The changes should be readily available on the webpage as codepipeline initiated another automatic deployment

Adding CodeBuild to the existing pipeline

Step 1: Edit the existing pipeline by clicking the edit button in the pipeline project

The screenshot shows the AWS CodePipeline console. The navigation path is: Developer Tools > CodePipeline > Pipelines > my-code-pipeline. Below the navigation, the pipeline name "my-code-pipeline" is displayed. A row of buttons includes "Notify" (with a dropdown arrow), "Edit" (highlighted in blue), "Stop execution", "Clone pipeline", and "Release change". The main content area shows a single stage named "Source" with status "Succeeded". It details "Pipeline execution ID: 51df86a5-181e-40b0-9f8b-691b86e924bf". The "Source" component is identified as "AWS CodeCommit" and was "Succeeded - Just now".

Step 2: Click Add Stage between the Source and Deploy stage.

The screenshot shows the AWS CodePipeline console with two stages visible: "Source" and "Deploy". The "Source" stage is currently selected, indicated by a grey background. The "Edit: Source" header is at the top, with an "Edit stage" button to its right. Below the stage name, it shows "Source" and "AWS CodeCommit". Between the Source and Deploy stages, there is a horizontal line with a plus sign and the text "+ Add stage". The "Deploy" stage below it has its own "Edit: Deploy" header and "Edit stage" button, showing "Deploy" and "AWS CodeDeploy".

Step 3: Enter a name for your Stage and click Add stage

Add stage

Stage name

No more than 100 characters

Cancel Add stage

Edit stage

Deploy

AWS CodeDeploy

+ Add stage

Step 4: Click Add action Group

Source



AWS CodeCommit

+ Add stage

Edit: my-first-webpage-build

Cancel

Delete

Done

+ Add action group

+ Add stage

Step 5: Enter the details as per your CodeBuild project

The screenshot shows the 'Edit action' configuration page for a new build action. The left sidebar lists navigation options: Pipeline, Action, Action provider, Region, Input artifacts, Project name, and Environment variables. The main form fields are:

- Action name:** MyBuild (input field)
- Action provider:** AWS CodeBuild (dropdown menu)
- Region:** Asia Pacific (Mumbai) (dropdown menu)
- Input artifacts:** SourceArtifact (dropdown menu), Add button, and a note about character limit.
- Project name:** my-code-build-project1 (input field with search icon and 'Create project' button).
- Environment variables - optional:** A note about choosing keys, values, and referencing variables generated by CodePipeline.

Step 6: Click Save

Step 6.1: Edit the Deploy Stage to have the source as TestOutput

Step 7: Click Release Change. This should start the build.

Step 8: If you get failed status click details under the error and go to phase details, you can see the error as shown below.

```
CLIENT_ERROR: AccessDenied: Access Denied status code: 403, request id: 848D104C8D350A28,  
host id:  
dzLeegcA8LQVpnNgxuUosAtQgtUw6TId1Ybhjgd4W8E8fnH4PnP4jjtUm12cE8QwfdWK/Rh6Jyk=  
for primary source and source version arn:aws:s3:::my-code-build-artifacts-bucket/my-code-  
pipeline/SourceArti/fgKY6Be
```

Step 9: Got to IAM → roles → search for codebuild and expand the codebuild rule

The screenshot shows the AWS IAM Roles page. At the top, there are tabs: Permissions (selected), Trust relationships, Tags, Access Advisor, and Revoke sessions. Below the tabs, it says "Permissions policies (1 policy applied)". There is a blue "Attach policies" button and a "Add inline policy" button. A table lists one policy: "CodeBuildBasePolicy-my-code-build-project..." which is a Managed policy.

Click JSON tab and edit policy

The screenshot shows the AWS IAM Roles page with the JSON tab selected. It displays the policy document in JSON format. The policy allows log writing to specific log groups.

```
1 {  
2     "Version": "2012-10-17",  
3     "Statement": [  
4         {  
5             "Effect": "Allow",  
6             "Resource": [  
7                 "arn:aws:logs:ap-south-1:039121988859:log-group:/aws  
8                     /codebuild/my-code-build-project1",  
9                 "arn:aws:logs:ap-south-1:039121988859:log-group:/aws  
10                    /codebuild/my-code-build-project1:/*"  
11             ]  
12         }  
13     ]  
14 }
```

Go to the rule where you have your bucket

Visual editor JSON

```
37
38  {
39    "Effect": "Allow",
40    "Resource": [
41      "arn:aws:s3:::my-code-build-artifacts-bucket",
42      "arn:aws:s3:::my-code-build-artifacts-bucket/*"
43    ],
44    "Action": [
45      "s3:PutObject",
46      "s3:GetBucketAcl",
47      "s3:GetBucketLocation"
```

Add a new Action “s3:GetObject” at the top as below and click Review and save .

```
RESOURCE . L
  "arn:aws:s3:::my-code-build-artifacts-bucket"
  "arn:aws:s3:::my-code-build-artifacts-bucket/"
],
"Action": [
  "s3:GetObject",
  "s3:PutObject",
  "s3:GetBucketAcl",
  "s3:GetBucketLocation"
]
},
```

Step 10: Go back to the pipeline and click retry on the build stage.

Step 11: You should see successful on all the phases.

Step 12: Go to the browser and type the Public IP to get a congratulations page which confirms the successful deployment

of your webapp using codepipeline (CodeCommit, CodeBuild, CodeDeploy)

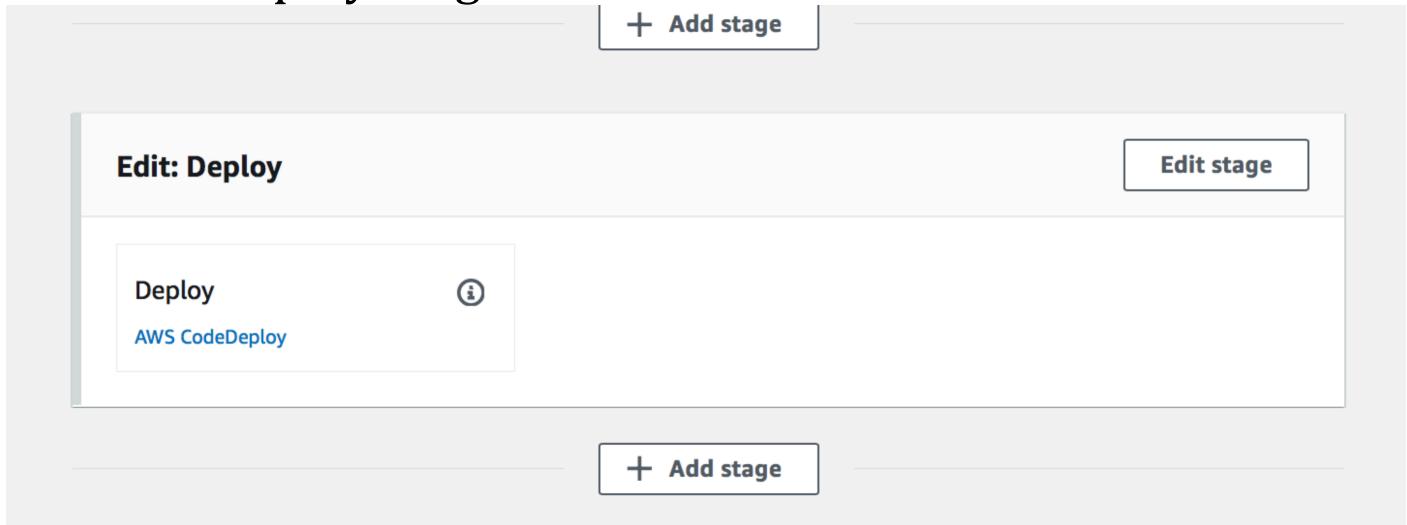
Congratulations

This application was deployed on i-0463e4e345f6af4f8 in ap-south-1a using AWS CodeDeploy.

For next steps, read the [AWS CodeDeploy Documentation](#).

Adding a new Production stage and adding a manual approval process

Step 1: Edit the existing code pipeline project and click Add Stage after the Deploy Stage



Step 2: Enter the stage name as ProdDeploy and click Add Stage

The screenshot shows the 'Edit: ProdDeploy' stage configuration screen. At the top right are 'Cancel', 'Delete', and 'Done' buttons. Below them is a 'Done' button. A large 'Add action group' button is visible. At the bottom center is a prominent 'Add stage' button.

Step 3: Click Add Sction Group and give an action name, Action Provider as “Manual Approval” and optianally a SNS topic if you have configured

The screenshot shows the 'Edit action' configuration screen. On the left, a vertical sidebar lists navigation items: Code Pipelines, Source, Build, Deploy, Pipelines, Getting Started, Pipelines, Pipelines, Pipelines, History, Settings, Go to, and Feedback. The main area contains the following fields:

- Action name: Approve Production Instances
- Action provider: Manual approval
- SNS topic ARN - optional: arn:aws:sns:ap-south-1:039121988859:mytopic
- URL for review - optional: (empty input field)
- Comments - optional: (empty input field)

Step 4: Add another action group below it, for production deployment. Select an action name, Action provider=CodeDeploy, InputArtifacts=ResultOutput(from code build stage), deploy application, and deployment = Your production deployment group that you should have in codedeploy module

Edit action

Action name
Choose a name for your action

No more than 100 characters

Action provider

Region

Input artifacts
Choose an input artifact for this action. [Learn more](#)

No more than 100 characters

Application name
Choose an application that you have already created in the AWS CodeDeploy console. Or create an application in the AWS CodeDeploy console and then return to this task.

Deployment group
Choose a deployment group that you have already created in the AWS CodeDeploy console. Or create a deployment group in the AWS CodeDeploy console and then return to this task.

Step 5: Click Done and Save and then click Release change to start the code pipeline. The pipeline execution should stop at the approval stage for review

Pipeline execution ID: [d7a20328-5796-4887-a969-33a2227ad9cf](#)

ApproveProductionIns...

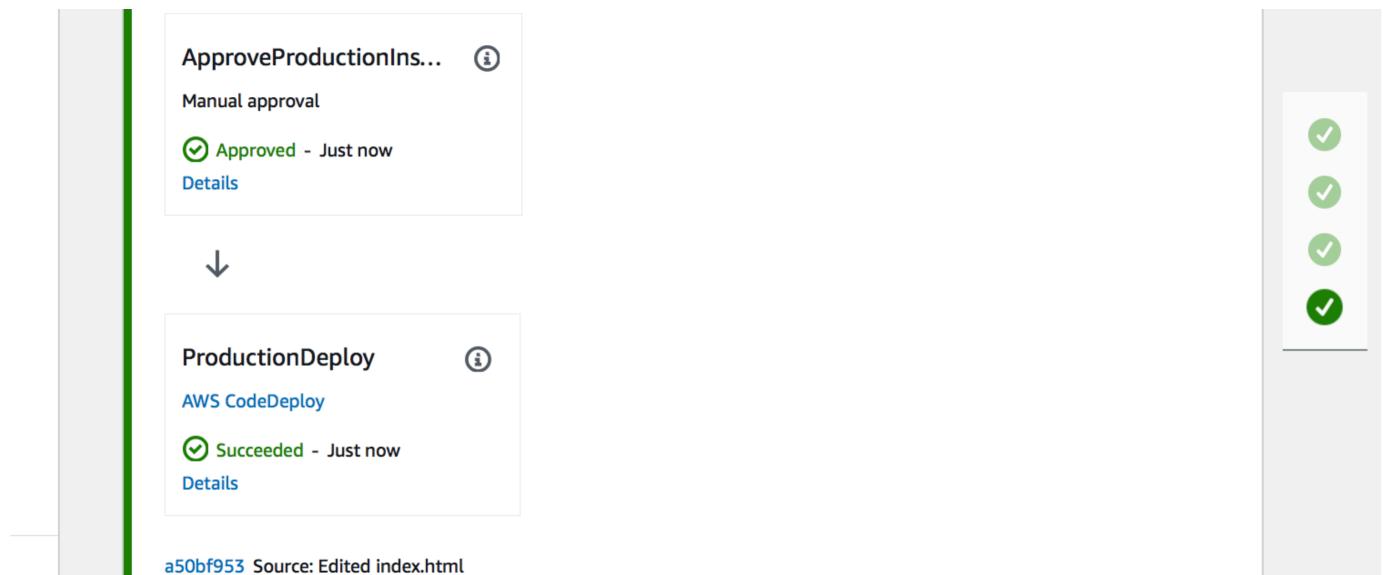
Manual approval

⌚ Waiting for approval -

↓

✓
✓
✓
⟳

Step 6: Click the Review button and provide approval comments to kick start the deployment to production instances.



Step 7: Test your application by pointing the production instance IP in the web page